CONTENTS

NASA Technical Paper 1814

Pitfalls and Guidelines for the Numerical Evaluation of Moderate-Order System Frequency Response

Harold P. Frisch

JUNE 1981

NASA

All measurement values are expressed in the International System of Units (SI) in accordance with NASA Policy Directive 2220.4, paragraph 4.

NASA Technical Paper 1814

Pitfalls and Guidelines for the Numerical Evaluation of Moderate-Order System Frequency Response

Harold P. Frisch
Goddard Space Flight Center
Greenbelt, Maryland

National Aeronautics
and Space Administration

**Scientific and Technical
Information Branch**

1981

ABSTRACT

The design and evaluation of a feedback control system via frequency response methods relies heavily upon numerical methods. In application, one can usually develop low-order simulation models which for the most part are devoid of numerical problems. However, when complex feedback interactions, for example, between instrument control systems and their flexible mounting structure, must be evaluated, simulation models become moderate to large order and numerical problems become common. The fundamental reason for this is that as system order enters the range of about 20 to 200, many popular tried and true methods are subject to severe numerical problems. An attempt is made in this paper to summarize a large body of relevant numerical error analysis literature in a language understandable to nonspecialists. The intent is to provide engineers using simulation models with an "engineering feel" for potential numerical problems without getting intertwined in the complexities of the associated mathematical theory. Guidelines are also provided by suggesting alternate state-of-the-art methods which have good numerical evaluation characteristics.

Blank Page

CONTENTS

PITFALLS AND GUIDELINES FOR THE NUMERICAL EVALUATION OF MODERATE-ORDER SYSTEM FREQUENCY RESPONSE

Harold P. Frisch
Goddard Space Flight Center
Greenbelt, Maryland

INTRODUCTION

The NASA large space system technology (LSST) program is attempting to develop an interactive integrated analysis capability (IAC). The objective is to provide engineers with an interdisciplinary analysis capability supportive of static, time, and frequency domain design and performance evaluation methods. In order to achieve this objective, standard analysis methods must be critically evaluated to determine their suitability for moderate- and large-order system response analysis. Unsuitable methods must be flagged and improved methods developed.

Most frequency domain analyses are adequately carried out with low- to moderate-order systems; i.e., systems of order 20 or less. Unfortunately, low- to moderate-order systems are not always adequate; for example, in assessing the cross-coupling effects between two or more on-board instrument control systems via spacecraft system flexibility. Systems such as this require moderate- to large-order system models. Within the context of this report, systems of order between about 20 and 200 are considered to be moderate to large order. In practical application, systems of larger order are usually not required if good engineering judgment is used in setting up the simulation model. When systems of an order higher than 200 are required and standard algorithms are used, resultant computing problems will normally be inordinate due to excessive run-time, memory requirements, and round-off error buildup. Solutions in these cases are usually only possible if specialized techniques are carefully used.

This paper is primarily concerned with the numerical methods used to set up and evaluate the "transfer function" of single variable feedback control theory and the "transfer function matrix" of multivariable feedback control theory. The remarks to be made apply to transfer functions written with respect to either the Laplace transform variable s (continuous time systems) or to the Z-transform variable z (discrete and sampled data systems).

It will be shown, by presenting the results of a few simple numerical experiments, that commonly used numerical methods, while excellent for low-order systems, begin to break down in an unpredictable manner as system order increases. To avoid this problem, alternate methods are presented which can be reliably used for moderate- to large-order systems.

It is assumed that transfer functions stem from a physically realizable, well-conditioned system. A well-conditioned system is defined as one for which slight perturbations in the system's physical characteristics (mass, geometry, feedback gains, etc.) produce correspondingly slight changes in resultant frequency response characteristics. An unstable system is a well-conditioned system if slight perturbations in its physical characteristics make it slightly more or less unstable. Ill-conditioned systems are of minimal importance in application, and therefore are not discussed herein. Ill-conditioned systems should be redesigned rather than analyzed.

The intent of this paper is to summarize, from the theory of condition, information which can be used to expose potential pitfalls, and to provide guidelines for selecting numerically reliable methods.

BACKGROUND

The historical starting point for the subject of numerical error analysis is usually marked by Reference 33, the 1947 von Neumann and Goldstine paper "Numerical Inverting of Matrices of High Order." It is an interesting historical note that in 1947 "high order" referred to matrices of order greater than or equal to 10, and "unusually large order" referred to order greater than 100 (Reference 33, pages 1022 and 1031). In addition to its historical interest, the discussion in its first chapter on the sources of error in a computation provides the framework from which a discussion of "condition" can conveniently begin. Four primary sources of error are enumerated: (1) Approximations implied by the mathematical model; (2) Errors in observational data, (3) Finitistic approximations to transcendental and implicit mathematical formulations; and (4) Errors associated with computing instruments (e.g., round-off errors in digital computation).

In their paper, von Neumann and Goldstine state that their follow-on development pertains only to errors of the fourth type. Their objective was to establish rigorous error estimates with respect to fixed point digital computation round-off error for the matrix inversion problem. This they do via a long and difficult development, which is reviewed in Reference 34 by Wilkinson. It is pointed out therein that the complexity of the von Neumann and Goldstine development effectively stymied follow-on research in numerical error analysis for the following decade. This situation existed until Givens, in Reference 13, recognized that round-off errors generated during the course of computation could be interpreted as equivalent perturbations on observational data. Furthermore, associated perturbation bounds could be established and computed quantities taken as exact solutions to the problem defined by the perturbed set of observational data. Nearly all recent analysis makes use of this approach. It has the advantage that round-off errors are placed on the same footing as errors in input data specification, the effect of which is usually considered during the course of a system analysis. This approach is generally referred to as backward error analysis.

In concept, "condition" starts with a computing problem. It then asks the following question: If the initial data to the computing problem is perturbed by a given amount, how much will the computed solution differ from the exact solution to the unperturbed problem? If small perturbations in

the initial data lead to correspondingly small perturbations in computed results the computing problem is considered to be "well-conditioned." If small perturbations in the initial data lead to large perturbations in computed results the computing problem is considered to be "ill-conditioned." Beyond this point, theory becomes complicated by subtle differences in approach and by the recognition that for multi-output computing problems; e.g. eigenanalysis, part of the output may be well-conditioned, while the rest may be either ill-conditioned or almost ill-conditioned.

A general theory of condition is presented in Reference 25 by Rice. In that reference, Rice attempts to analyze the way in which an uncertainty (perturbation) in the initial data propagates to an uncertainty in the solution of the system of equations. While the results presented by Rice can be applied to computational problems, they cannot be construed to form a theory of the condition of a computation (see Reference 25, page 288).

Before one erroneously links round-off error and ill-conditioning together, it is important to point out that ill-conditioning exists independently of round-off error effects and that its intrinsic nature is unaffected by various schemes for round-off error control. The link between round-off error and ill-conditioning comes about after the decision is made to view the accumulated effects of round-off error as a perturbation on observational data.

Wilkinson applies this point of view in his discussions on ill-conditioned problems in Reference 35. On page 28 he defines a computing problem as ill-conditioned if very small relative perturbations in the parameters make comparatively large errors in the solutions.

Steward, in Reference 28, page 76, defines a "stable algorithm" as an algorithm which yields a solution which is near the exact solution to the slightly perturbed problem. For ill-conditioned problems, the exact solution of the perturbed problem may not be near the exact solution of the unperturbed problem and thus, even though a stable algorithm is used to solve an ill-conditioned problem, the computed solution and the exact solution need not agree at all. When applied to a well-conditioned problem, errors introduced by the stable algorithm are no more than are warranted by the data.

The motivation for this paper stems primarily from the need to obtain transfer functions of moderate- to large-order systems and the following statement by Wilkinson in Reference 34, page 565. In effect it says that certain instinctive methods for setting up and evaluating transfer functions should be avoided:

> "Perhaps the most important lesson that has been learned is the rather negative one, that in general it is undesirable to reduce implicit polynomial equations, particularly determinantal equations, to explicit polynomial form. Such a reduction is superficially attractive both because of the formal simplicity of the explicit form and because there is a natural tendency to think that 'we know all about polynomials.' Commonly a catastrophic worsening of the condition of the problem takes place."

The objective of this paper is to explore the dilemma brought about by the fact that the instinctive and most commonly used method for obtaining transfer functions is "by reducing implicit polynomial equations, particularly determinantal equations, to explicit polynomial form." This exploration is to be done via numerical examples designed to provide the engineer with some insight as to how far instinctive methods can be reliably used, and then to define what should be done when the instinctive methods begin to numerically fail.

POLYNOMIAL AND RATIONAL FORMS FOR TRANSFER FUNCTIONS

Virtually all frequency-response methods (see Reference 19) require transfer functions to be defined as rational functions in the form:

$$R(x) = \frac{P_m(x)}{Q_n(x)} \tag{1}$$

where the degree m of the numerator polynomial $P_m(x)$ is less than or equal to the degree n of the denominator polynomial $Q_n(x)$. Multivariable methods require transfer function matrices to be defined as rectangular matrices of rational functions. These normally are expressed in the form:

$$G(x) = \frac{N(x)}{d(x)} \tag{2}$$

where N(x) is a rectangular matrix of polynomials and d(x) is the polynomial which is the least common denominator of all elements in G(x).

If the frequency response analysis is pursued analytically, the polynomials contained in equations 1 and 2 may be expressed in a variety of equivalent forms, whichever is most convenient for the follow-on analytic development. However, if the need for numerical evaluation is on the bottom line, it must be recognized that the evaluation properties of equivalent polynomial forms differ drastically relative to each other. (Refer to Rice, Reference 24 and Gautschi, Reference 12 for an in-depth study.) In summary, Rice, Gautschi, Wilkinson (in Reference 35), and several other authors conclude that polynomials expressed in the "root product" form, that is:

$$P_n(x) = A_n \prod_{i=1}^{n} (x - z_i) \tag{3}$$

are comparatively well-conditioned relative to polynomials expressed in the "nested" or "power series" form, that is:

$$P_n(x) = (\cdots((A_n x + A_{n-1}) x + A_{n-2}) x + \cdots + A_1) x + A_0 \qquad (4)$$
$$= \sum_{i=0}^{n} A_i x^i$$

For frequency response analysis, this is not the end (see Wilkinson, Reference 35, page 47). In nearly all frequency response problems, neither the zeros of equation (3) nor the coefficients of equation (4) can be considered as primary system definition data. Therefore, the problem of determining whether or not these parameters have been evaluated with an accuracy sufficient for the numerical evaluation of the system's "true" frequency response characteristics must also be considered.

The following portion of this section explores this problem by presenting the results of a numerical experiment. The experiment and data presentation format has been structured to provide insight; the actual numeric magnitudes are of minimal importance. In fact, if the experiment were to be repeated on a different computer with different precision, a one-for-one correlation would not be expected. The important point is that computational problems can develop rapidly, as system order increases for certain classes of problems.

Methods used to compute roots for the root product form will not be explored. If roots are obtained via the EISPACK eigenanalysis methods provided in Reference 27 and if primary system definition data is used to define associated matrix coefficients, then, since all algorithms given therein are stable, errors involved in the computation of eigenvalues are no more than are warranted by the system definition data. Refer to the relevant papers collected in Reference 36 for details; nearly all EISPACK routines are based upon the ALGOL procedures presented and developed therein.

The power series form for polynomials is a favorite in controls analysis. Coefficients are easily computed and zeros obtainable via application of a variety of stable, well-established polynomial root-finding algorithms; e.g. RPOLY (Reference 15). Unfortunately, the use of a stable algorithm is no guaranty that accurate results will be obtained for an ill-conditioned computing problem. As system order increases, polynomial coefficients can only be computed with a finite degree of accuracy on any computing system, i.e., to machine precision. This limit on achievable accuracy, when viewed as a perturbation on input data, leads to errors in computed results which far exceed that warranted by the imprecision associated with the computed polynomial coefficients.

In order to illustrate this point in a manner which will provide some physical intuition, two test problems are chosen: first, a hypothetical non-oscillatory system; and second, a hypothetical damped oscillatory system.

Non-Oscillatory System

Let:

$$(X + \lambda_1)(X + \lambda_2)(\cdots)(X + \lambda_N) = 0 \tag{5}$$

where $\lambda_i = 1, 2, \cdots, N$ define the first system.

Damped Oscillatory System

Let:

$$(X^2 + 2\zeta\omega_1 X + \omega_1^2)(X^2 + 2\zeta\omega_2 X + \omega_2^2)(\cdots)(X^2 + 2\zeta\omega_N X + \omega_N^2) = 0 \tag{6}$$

where $\omega_i = 1, 2, \cdots, N$ and $0 \leqslant \zeta \leqslant 1$ define the second.

In actuality, many values of ζ have been examined, however results are only presented for ζ = .0001, .9, and 1.0. For insight purposes, this is sufficient. Results for other values of ζ are obtainable, approximately, from an exponential curve fitted between the three relevant points provided.

Procedure

Step 1. Utilizing floating point double precision arithmetic, compute the coefficients to the power series equivalents of equations 5 and 6. Use N = 2, 3, $\cdots$, 25 for equation 5, and N = 2, 3, $\cdots$, 19 for equation 6 and the values of ζ stated above.

Step 2. Transfer with full precision the coefficients for each polynomial into the zero-finding algorithm. The algorithm used was an in-house variation of the popular Jenkins and Traub algorithm RPOLY given in Reference 15 and discussed in Reference 16. To check root quality, all inputted coefficients must be reproducible to 7 digits accuracy with the computed zeros. If not, an error flag is automatically set prior to subroutine exit. In all cases presented herein, successful quality checks were obtained.

Step 3. Compute the number of digits in agreement between computed and exact solutions via the equation:

$$d = -\log_{10}\left(\frac{e - c}{e}\right) \tag{7}$$

where:

e = exact solution
c = computed solution
d = number of digits in agreement

All computed zeros are arranged in increasing order of magnitude and compared with the exact solutions which are also arranged in increasing order of magnitude.

All results are presented in Tables 1 through 8, in matrix format. The element d_{nj} in row n column j is to be interpreted as meaning that "the j-th zero in increasing order of magnitude of the polynomial of degree n has been computed with d_{nj} digits of accuracy." If $d_{nj} \geq 15.7$ this is to be interpreted to imply exact agreement. All results were obtained from an IBM 360-91, using 64 bits for double precision computation.

A full set of analogous results was also obtained by using the EISPACK (Reference 27) eigenvalue finding algorithms. In theory, the eigenvalues of the companion matrix (Frobenius form) to the associated characteristic polynomial equal the zeros of the polynomial. In actuality, eigenvalues computed via the QR-algorithm were inferior in quality to the zeros computed via the polynomial root-finding algorithm. However, they were surprisingly good, considering the enormous spread in magnitude of the non-zero elements in the companion matrix. This observation is in agreement with round-off error analysis theory (see Reference 37, pages 432-434 and Reference 14, pages 147-148).

Tables 1 and 2 present results associated with equation 5. This is a favorite example in the literature; it has been studied by Gautschi (Reference 12, page 410) and Wilkinson (Reference 37, page 418). In Table 1 it is noted that there is a steady deterioration in compatational accuracy, as system order increases from 2 through 25. Twenty (20) marks the point at which computed results deviate significantly enough from the exact for false conclusions to be a real problem. Up to N = 22, all computed roots are real, as they should be, as shown in Table 2. For N = 23 and beyond, some adjacent real roots are computed incorrectly as complex pairs. This phenomenon is well-known, and reported also in Reference 37.

Tables 3 and 4 present results associated with equation 6 for $\zeta = .0001$. This is the case of a nearly undamped system. It is evident from the results presented that both real and imaginary parts of all zeros are computed to a surprisingly high degree of accuracy for polynomials up to degree 38. An important point which is not obvious from the tables is that in all cases the real part of each zero is computed with the correct sign.

Tables 5 and 6 present results associated with equation 6 for $\zeta = .90$. These results are intended to provide data at the approximate point at which a breakdown in computational ability becomes significant, that is, when system roots approach coincidence in the complex plane or, in engineering terms, when the damping associated with system frequencies approaches "critical."

Table 1 – Root 1, 2, . . ., N

MATRIX ELEMENTS = -DLOG10((EXACT - COMPUTED)/EXACT); REAL PARTS OF POLYRT COMPUTED ROOTS

		(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)	(11)	(12)	(13)	(14)	(15)	(16)	(17)	(18)	(19)	(20)
2	1	15.7	15.7																		
3	1	15.7	15.7	15.3																	
4	1	15.1	14.6	14.5	15.0																
5	1	15.7	14.9	14.7	15.6	15.2															
6	1	15.2	13.8	13.1	12.7	12.7	13.2														
7	1	15.2	13.3	12.6	12.2	12.1	12.2	12.8													
8	1	14.4	13.6	13.0	12.5	12.6	12.9	12.5	12.9												
9	1	15.4	14.1	12.6	11.8	11.2	10.9	10.8	11.0	11.6											
10	1	15.8	13.7	12.4	11.5	10.9	10.6	10.4	10.5	10.9	11.6										
11	1	15.0	13.3	11.9	11.0	10.4	10.0	9.7	9.6	9.7	10.1	10.8									
12	1	14.9	12.7	11.3	10.3	9.6	9.1	8.8	8.6	8.7	8.9	9.4	10.3								
13	1	15.0	12.8	11.4	10.5	9.9	9.6	9.9	8.8	8.4	8.4	8.6	9.0	9.9							
14	1	14.6	12.6	11.1	10.0	9.1	8.5	8.1	7.8	7.8	8.0	9.1	8.6	8.8	9.6						
15	1	14.9	12.4	11.1	10.0	9.2	8.7	8.4	9.4	7.8	7.4	7.3	7.4	7.8	8.3	9.2					
16	1	14.5	12.5	11.0	9.9	8.9	8.1	7.5	7.1	6.8	6.7	6.7	6.9	7.2	7.5	8.0	8.8				
17	1	14.7	12.9	10.8	9.4	8.4	7.5	6.8	6.3	5.9	5.6	5.4	5.3	5.4	5.7	6.2	6.8	7.9			
18	1	14.4	12.5	10.9	10.0	8.7	7.3	6.3	5.6	5.0	4.6	4.3	4.1	4.1	4.2	4.4	4.8	5.5	6.4		
19	1	14.5	12.2	10.5	9.1	8.0	7.1	6.4	5.9	5.6	5.4	5.2	4.9	4.5	4.3	4.3	4.5	4.8	5.3	6.3	
20	1	14.4	12.3	9.7	8.6	7.7	7.0	6.4	5.8	5.2	4.8	4.5	4.7	4.3	3.7	3.5	3.5	3.6	4.0	4.5	5.5
21 21	1 21	13.8 4.7	11.6	9.4	8.1	6.9	6.0	5.1	4.4	3.8	3.2	2.8	2.4	2.2	2.0	2.0	2.0	2.2	2.6	3.0	3.7
22 22	1 21	14.5 3.4	11.6 4.4	9.3	8.0	6.8	5.9	5.3	4.9	4.7	3.6	3.0	2.7	2.7	4.9	2.3	1.9	1.9	2.0	2.2	2.7
23 23	1 21	13.6 1.8	12.0 2.6	9.3 3.5	7.9	6.6	5.5	4.5	3.7	3.0	2.4	2.0	1.5	1.3	1.6	1.4	1.5	1.5	1.5	1.7	1.5
24 24	1 21	14.1 2.2	12.5 2.1	10.5 2.6	7.9 3.5	6.5	5.3	4.3	3.5	2.8	2.3	1.8	1.7	2.1	1.8	1.3	1.6	1.5	1.4	1.3	1.8
25 25	1 21	14.0 1.4	11.4 2.3	9.7 1.5	7.9 1.9	6.5 3.2	5.4	4.4	3.6	3.0	2.4	1.7	1.4	1.8	1.3	1.9	1.3	1.6	1.5	1.4	1.8

MATRIX ELEMENTS = -DLOG10((EXACT - COMPUTED)/EXACT); IMAGINARY PARTS OF POLYRT COMPUTED ROOTS

		(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)	(11)	(12)	(13)	(14)	(15)	(16)	(17)	(18)	(19)	(20)
2	1	15.7	15.7																		
3	1	15.7	15.7	15.7																	
4	1	15.7	15.7	15.7	15.7																
5	1	15.7	15.7	15.7	15.7	15.7															
6	1	15.7	15.7	15.7	15.7	15.7	15.7														
7	1	15.7	15.7	15.7	15.7	15.7	15.7	15.7													
8	1	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7												
9	1	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7											
10	1	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7										
11	1	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7									
12	1	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7								
13	1	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7							
14	1	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7						
15	1	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7					
16	1	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7				
17	1	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7			
18	1	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7		
19	1	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	
20	1	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7
21	1	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7
21	21	15.7																			
22	1	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7
22	21	15.7	15.7																		
23	1	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	0.5	0.5	0.1	0.1	0.0	0.0	0.1	0.1	0.5
23	21	0.5	15.7	15.7																	
24	1	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	0.2	0.2	0.0	0.0	0.0	0.0	15.7	
24	21	15.7	15.7	15.7	15.7																
25	1	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	0.1	0.1	0.0	0.0	0.0	0.0	0.0	0.0	
25	21	0.0	0.0	0.0	0.0	15.7															

Table 3 – ζ = .0001

MATRIX ELEMENTS = -DLOG10((EXACT - COMPUTED)/EXACT); REAL PARTS OF POLYRT COMPUTED ROOTS

		(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)	(11)	(12)	(13)	(14)	(15)	(16)	(17)	(18)	(19)	(20)
4	1	15.9	15.9	15.9	15.9																
6	1	16.2	16.2	11.4	11.4	11.5	11.5														
8	1	15.7	15.7	11.9	11.9	11.8	11.8	12.3	12.3												
10	1	15.5	15.5	12.2	12.2	12.0	12.0	12.3	12.3	12.9	12.9										
12	1	15.6	15.6	12.4	12.4	12.1	12.1	12.2	12.2	12.7	12.7	13.5	13.5								
14	1	14.8	14.8	12.5	12.5	12.2	12.2	12.3	12.3	12.5	12.5	11.0	11.0	11.1	11.1						
16	1	15.5	15.5	12.6	12.6	12.3	12.3	12.3	12.3	12.6	12.6	9.2	9.2	9.0	9.0	9.4	9.4				
18	1	15.1	15.1	12.7	12.7	12.3	12.3	12.2	12.2	12.5	12.5	9.0	9.0	8.7	8.7	8.8	8.8	9.4	9.4		
20	1	15.1	15.1	12.7	12.7	12.3	12.3	12.4	12.4	12.9	12.9	9.2	9.2	8.8	8.8	8.9	8.9	9.4	9.4	10.5	10.6
22	1	15.3	15.3	12.7	12.7	12.3	12.3	12.1	12.1	11.7	11.7	9.3	9.3	8.9	8.9	8.8	8.8	9.1	9.1	9.7	9.7
22	21	10.9	10.9																		
24	1	15.1	15.1	12.8	12.8	12.4	12.4	12.1	12.1	11.5	11.5	9.5	9.5	8.4	8.4	8.0	8.0	7.9	7.9	8.0	8.0
24	21	8.4	8.4	9.2	9.2																
26	1	15.1	15.1	12.8	12.8	12.4	12.4	12.4	12.4	12.2	12.2	9.6	9.6	8.0	8.0	7.5	7.5	7.2	7.2	7.3	7.3
26	21	7.6	7.6	8.1	8.1	9.0	9.0														
28	1	15.3	15.3	12.8	12.8	12.4	12.4	12.7	12.7	11.2	11.2	9.6	9.6	7.8	7.8	7.1	7.1	6.8	6.8	6.8	6.8
28	21	7.0	7.0	7.7	7.7	8.0	8.0	8.3	8.3												
30	1	15.2	15.2	12.9	12.9	12.6	12.6	12.2	12.2	11.1	11.1	9.6	9.6	7.6	7.6	6.9	6.9	6.6	6.6	6.5	6.5
30	21	6.6	6.6	6.8	6.8	7.3	7.3	7.9	7.9	8.8	8.8										
32	1	15.4	15.4	12.8	12.8	12.4	12.4	12.7	12.7	11.2	.2	9.6	9.6	7.5	7.5	6.7	6.7	6.4	6.4	6.2	6.2
32	21	6.2	6.2	6.4	6.4	6.8	6.8	7.3	7.3	8.0	.0	8.9	8.9								
34	1	15.2	15.2	12.8	12.8	12.5	12.5	12.1	12.1	11.0	11.0	9.7	9.7	7.4	7.4	6.6	6.6	6.2	6.2	6.0	6.0
34	21	6.0	6.0	6.1	6.1	6.6	6.6	7.0	7.0	6.8	6.8	7.1	7.1	7.9	7.9						
36	1	15.1	15.1	12.8	12.8	12.3	12.3	12.8	12.8	11.1	11.1	9.8	9.8	7.3	7.3	6.5	6.5	6.1	6.1	5.8	5.8
36	21	5.8	5.8	5.9	5.9	6.1	6.1	6.5	6.5	7.7	7.7	7.2	7.2	7.4	7.4	8.2	8.2				
38	1	15.2	15.2	12.9	12.9	13.2	13.2	11.8	11.8	11.3	11.3	10.1	10.1	7.2	7.2	6.4	6.4	6.0	6.0	5.7	5.7
38	21	5.7	5.7	7.2	7.2	5.4	5.4	5.1	5.1	5.1	5.1	5.4	5.4	5.6	5.8	6.4	6.4	7.5	7.5		

Table 4 – $\zeta = .0001$

MATRIX ELEMENTS = -DLOG10((EXACT - COMPUTED)/EXACT); IMAGINARY PARTS OF POLYRT COMPUTED ROOT

		(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)	(11)	(12)	(13)	(14)	(15)	(16)	(17)	(18)	(19)	(20)
4	1	15.7	15.7	15.7	15.7																
6	1	15.7	15.7	15.7	15.7	15.7	15.7														
8	1	15.7	15.7	16.0	16.0	15.5	15.5	15.7	15.7												
10	1	15.7	15.7	15.7	15.7	15.1	15.1	15.6	15.6	15.9	15.9										
12	1	15.7	15.7	15.3	15.3	14.9	14.9	15.7	15.7	14.8	14.8	15.1	15.1								
14	1	15.7	15.7	15.2	15.2	14.5	14.5	14.2	14.2	14.0	14.0	14.0	14.0	14.5	14.5						
16	1	15.7	15.7	14.8	14.8	14.5	14.5	14.5	14.5	13.8	13.8	13.4	13.4	13.4	13.4	13.8	13.8				
18	1	15.7	15.7	14.8	14.8	13.9	13.9	13.3	13.3	12.9	12.9	12.6	12.6	12.6	12.6	12.8	12.8	13.5	13.5		
20	1	15.4	15.4	15.7	15.7	14.0	14.0	13.3	13.3	13.1	13.1	13.4	13.4	12.3	12.3	12.1	12.1	12.3	12.3	12.9	12.9
22	1	15.2	15.2	14.7	14.7	14.7	14.7	13.7	13.7	12.8	12.8	12.2	12.2	12.0	12.0	12.0	12.0	12.1	12.1	12.4	12.4
22	21	13.0	13.0																		
24	1	15.2	15.2	14.8	14.8	14.2	14.2	13.6	13.6	12.5	12.5	11.9	11.9	11.6	11.6	11.5	11.5	11.6	11.6	11.8	11.8
24	21	12.2	12.2	13.1	13.1																
26	1	15.4	15.4	14.7	14.7	14.5	14.5	13.1	13.1	12.4	12.4	11.8	11.8	11.5	11.5	11.6	11.6	11.9	11.9	11.1	11.1
26	21	10.9	10.9	11.2	11.2	11.8	11.8														
28	1	15.4	15.4	15.0	15.0	13.7	13.7	12.9	12.9	12.2	12.2	11.6	11.6	11.2	11.2	10.8	10.8	10.4	10.4	10.2	10.2
28	21	10.2	10.2	10.4	10.4	10.8	10.8	11.6	11.6												
30	1	15.4	15.4	15.2	15.2	13.9	13.9	13.5	13.5	13.2	13.2	12.5	12.5	11.3	11.3	10.5	10.5	10.0	10.0	9.7	9.7
30	21	9.8	9.8	9.7	9.7	10.0	10.0	10.4	10.4	11.2	11.2										
32	1	15.4	15.4	14.6	14.6	14.1	14.1	13.6	13.6	12.3	12.3	11.6	11.6	11.1	11.1	10.7	10.7	10.3	10.3	9.8	9.8
32	21	9.5	9.5	9.4	9.4	9.5	9.5	9.9	9.9	10.6	10.6	12.8	12.8								
34	1	15.4	15.4	14.7	14.7	14.1	14.1	13.4	13.4	12.3	12.3	11.8	11.8	11.5	11.5	10.3	10.3	9.7	9.7	9.4	9.4
34	21	9.3	9.3	9.4	9.4	9.9	9.9	10.0	10.0	9.9	9.9	10.2	10.2	11.0	11.0						
36	1	15.7	15.7	14.5	14.5	13.6	13.6	13.9	13.9	12.4	12.4	14.5	14.5	10.8	10.8	9.9	9.9	9.4	9.4	9.0	9.0
36	21	8.8	8.8	8.8	8.8	9.0	9.0	9.3	9.3	9.9	9.9	10.6	10.6	11.2	11.2	12.0	12.0				
38	1	15.7	15.7	14.4	14.4	13.5	13.5	13.0	13.0	12.8	12.8	12.2	12.2	11.1	11.1	10.6	10.6	10.9	10.9	9.4	9.4
38	21	8.8	8.8	8.8	8.8	8.5	8.5	8.8	8.8	9.5	9.5	8.9	8.9	9.0	9.0	9.4	9.4	10.3	10.3		

Table 5 – $\zeta = .9$

MATRIX ELEMENTS = -DLOG10((EXACT - COMPUTED)/EXACT); REAL PARTS OF POLYNT COMPUTED ROOTS

		(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)	(11)	(12)	(13)	(14)	(15)	(16)	(17)	(18)	(19)	(20)
4	1	15.9	15.9	15.9	15.9																
6	1	15.1	15.1	14.7	14.7	14.8	14.8														
8	1	15.3	15.3	13.6	13.6	13.5	13.5	13.4	13.4												
10	1	14.7	14.7	13.3	13.3	12.3	12.3	12.4	12.4	13.5	13.5										
12	1	14.2	14.2	12.9	12.9	11.6	11.6	11.5	11.5	11.3	11.3	12.1	12.1								
14	1	14.1	14.1	12.7	12.7	11.2	11.2	11.3	11.3	11.0	11.0	10.9	10.9	11.7	11.7						
16	1	13.9	13.9	11.6	11.6	10.6	10.6	9.5	9.5	9.2	9.2	9.1	9.1	10.1	10.1	10.4	10.4				
18	1	13.2	13.2	11.7	11.7	9.7	9.7	9.5	9.5	8.4	8.4	9.0	9.0	8.5	8.5	9.5	9.5	9.6	9.6		
20	1	13.7	13.7	11.9	11.9	9.9	9.9	8.5	8.5	8.2	8.2	7.2	7.2	7.6	7.6	7.4	7.4	7.7	7.7	8.5	8.6
22	1	13.6	13.6	10.5	10.5	9.2	9.2	8.4	8.4	7.6	7.6	6.7	6.7	7.2	7.2	6.4	6.4	6.8	6.8	7.3	7.8
22	21	8.7	8.7																		
24	1	13.4	13.4	10.4	10.4	8.7	8.7	7.4	7.4	6.6	6.6	6.1	6.1	5.6	5.6	6.1	6.1	5.6	5.6	6.0	6.0
24	21	6.7	6.7	7.1	7.1																
26	1	12.9	12.9	10.0	10.0	8.3	8.3	6.8	6.8	5.8	5.8	5.2	5.2	4.7	4.7	4.2	4.2	4.6	4.6	4.7	4.7
26	21	4.7	4.7	5.3	5.3	6.5	6.5														
28	1	13.2	13.2	10.0	10.0	8.2	8.2	6.8	6.8	5.5	5.5	5.3	5.3	4.2	4.2	4.5	4.5	4.1	4.1	3.9	3.9
28	21	4.1	4.1	4.5	4.5	5.4	5.4	7.0	7.0												
30	1	12.8	12.8	9.6	9.6	7.9	7.9	6.3	6.3	5.1	5.1	4.1	4.1	3.9	3.9	2.8	2.8	2.7	2.7	2.5	2.5
30	21	2.3	2.3	2.7	2.7	3.5	3.5	3.6	3.6	4.5	4.5										
32	1	12.6	12.6	9.5	9.5	7.6	7.6	7.9	7.9	4.6	4.6	3.7	3.7	2.9	2.9	2.4	2.4	2.0	2.0	2.2	2.2
32	21	1.7	1.7	2.3	2.3	2.4	2.4	2.8	2.8	4.1	4.1	4.8	4.8								
34	1	14.0	14.0	9.5	9.5	7.3	7.3	5.6	5.6	6.0	6.0	3.2	3.2	3.0	3.0	2.3	2.3	1.4	1.4	1.7	1.7
34	21	1.3	1.3	1.2	1.2	2.3	2.3	1.1	1.1	1.7	1.7	2.1	2.1	3.7	3.7						
36	1	12.6	12.6	10.7	10.7	7.1	7.1	5.5	5.5	4.2	4.2	3.1	3.1	3.1	3.1	1.9	1.9	1.2	1.2	1.3	1.3
36	21	0.9	0.9	1.0	1.0	1.4	1.4	1.8	1.8	1.2	1.2	2.6	2.6	1.8	1.8	2.6	2.6				

Table 6 – $\zeta = .9$

MATRIX ELEMENTS = -DLOG10((EXACT - COMPUTED)/EXACT); IMAGINARY PARTS OF POLYRT COMPUTED ROOT

		(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)	(11)	(12)	(13)	(14)	(15)	(16)	(17)	(18)	(19)	(20)
4	1	15.3	15.3	15.4	15.4																
6	1	14.7	14.7	14.3	14.3	14.9	14.9														
8	1	14.4	14.4	13.4	13.4	12.9	12.9	14.1	14.1												
10	1	15.5	15.5	12.6	12.6	12.4	12.4	12.0	12.0	12.4	12.4										
12	1	14.5	14.5	12.2	12.2	11.5	11.5	10.9	10.9	12.2	12.2	11.7	11.7								
14	1	13.3	13.3	11.8	11.8	12.0	12.0	10.7	10.7	10.7	10.7	11.3	11.3	11.3	11.3						
16	1	13.4	13.4	11.4	11.4	9.9	9.9	9.2	9.2	8.8	8.8	9.2	9.2	9.1	9.1	9.9	9.9				
18	1	13.6	13.6	10.9	10.9	9.8	9.8	8.5	8.5	8.7	8.7	8.0	8.0	9.2	9.2	8.5	8.5	9.2	9.2		
20	1	13.6	13.6	11.2	11.2	9.5	9.5	8.5	8.5	7.4	7.4	8.0	8.0	6.8	6.8	7.0	7.0	9.6	9.6	8.4	8.4
22	1	13.0	13.0	11.8	11.8	9.1	9.1	8.4	8.4	7.0	7.0	6.7	6.7	6.1	6.1	6.6	6.6	6.7	6.7	7.0	7.0
22	21	8.0	8.0																		
24	1	13.2	13.2	10.4	10.4	8.8	8.8	9.0	9.0	6.3	6.3	5.6	5.6	5.7	5.7	5.2	5.2	5.8	5.8	5.5	5.5
24	21	5.9	5.9	7.1	7.1																
26	1	12.4	12.4	9.9	9.9	8.1	8.1	6.8	6.8	5.6	5.6	4.7	4.7	4.3	4.3	4.5	4.5	3.9	3.9	4.0	4.0
26	21	4.8	4.8	5.5	5.5	6.1	6.1														
28	1	12.2	12.2	9.6	9.6	7.9	7.9	6.5	6.5	7.6	7.6	4.4	4.4	4.6	4.6	3.6	3.6	3.6	3.6	4.1	4.1
28	21	4.6	4.6	4.3	4.3	4.7	4.7	5.6	5.6												
30	1	12.2	12.2	10.1	10.1	7.4	7.4	5.9	5.9	4.8	4.8	4.0	4.0	3.0	3.0	3.4	3.4	2.2	2.2	2.0	2.0
30	21	3.2	3.2	2.2	2.2	2.5	2.5	3.4	3.4	5.0	5.0										
32	1	12.0	12.0	10.0	10.0	7.3	7.3	5.5	5.5	4.5	4.5	3.4	3.4	2.6	2.6	2.0	2.0	2.0	2.0	1.5	1.5
32	21	1.8	1.8	1.5	1.5	2.0	2.0	3.0	3.0	3.3	3.3	4.6	4.6								
34	1	12.0	12.0	9.8	9.8	7.9	7.9	5.4	5.4	3.8	3.8	3.5	3.5	2.3	2.3	1.5	1.5	1.4	1.4	0.8	0.8
34	21	0.3	0.3	0.9	0.9	0.8	0.8	0.9	0.9	1.2	1.2	1.9	1.9	2.6	2.6						
36	1	12.1	12.1	10.4	10.4	7.2	7.2	4.9	4.9	3.6	3.6	3.5	3.5	4.6	4.6	1.2	1.2	2.1	2.1	0.9	0.9
36	21	0.3	0.3	0.9	0.9	0.8	0.8	0.7	0.7	0.7	0.7	1.1	1.1	2.2	2.2	3.3	3.3				

Table 7 – $\xi = 1.0$

MATRIX ELEMENTS = -DLOG10((EXACT - COMPUTED)/EXACT); REAL PARTS OF POLYRT COMPUTED ROOTS

		(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)	(11)	(12)	(13)	(14)	(15)	(16)	(17)	(18)	(19)	(20)
4	1	8.3	8.3	15.7	15.7																
6	1	8.3	8.3	7.9	7.9	7.3	7.3														
8	1	8.4	8.4	7.3	7.3	7.7	7.7	7.8	7.8												
10	1	8.4	8.4	8.2	8.2	5.5	5.5	10.3	10.3	10.5	10.5										
12	1	8.4	8.4	5.4	5.4	5.1	5.1	5.1	5.1	4.8	4.8	5.0	5.0								
14	1	8.5	8.5	5.4	5.4	8.6	8.6	7.7	7.7	8.1	8.1	7.8	7.8	8.5	8.5						
16	1	8.5	8.5	5.4	5.4	4.3	4.3	3.7	3.7	3.3	3.3	3.3	3.3	3.5	3.5	4.1	4.1				
18	1	8.5	8.5	5.4	5.4	4.2	4.2	3.4	3.4	3.0	3.0	2.9	2.9	5.2	5.2	5.8	5.8	6.1	6.1		
20	1	8.5	8.5	5.4	5.4	3.7	3.7	2.9	2.9	2.5	2.4	2.5	2.5	3.6	3.6	4.0	4.0	3.8	3.8	4.9	4.9
22	1	8.5	8.5	5.4	5.4	3.8	3.8	2.8	2.8	2.3	2.2	3.0	3.0	2.4	2.4	2.8	2.8	2.5	2.5	2.7	2.7
22	21	3.6	3.6																		
24	1	8.5	8.5	5.3	5.3	3.1	3.1	2.2	2.1	1.5	1.1	1.0	1.3	1.0	1.3	1.1	1.3	1.2	1.2	1.4	1.1
24	21	1.6	1.2	1.7	1.9																
26	1	8.6	8.6	5.3	5.3	3.0	3.0	2.1	2.0	1.5	1.2	1.4	1.4	1.2	1.2	1.4	1.4	2.0	2.0	1.7	1.7
26	21	1.4	1.4	1.5	1.5	2.1	2.1														
28	1	8.6	8.6	5.4	5.4	2.8	2.7	1.7	1.4	1.0	1.7	0.8	2.1	0.8	2.5	0.9	1.7	1.0	1.3	1.3	1.1
28	21	1.7	1.0	2.4	1.0	1.9	1.1	2.8	1.8												
30	1	8.6	8.6	5.2	5.2	2.5	2.5	1.5	1.2	0.8	2.3	0.8	1.5	0.8	1.6	0.8	4.2	1.0	1.2	1.2	1.2
30	21	1.9	1.9	1.4	1.4	1.1	1.1	1.1	1.1	1.2	1.3										
32	1	8.3	8.3	5.2	5.2	2.5	2.5	1.4	1.3	0.8	1.6	0.7	1.2	0.7	1.1	0.7	1.2	0.8	1.5	0.9	3.5
32	21	1.0	1.9	1.1	1.2	1.7	0.9	1.4	0.8	1.1	0.3	1.1	1.0								
34	1	8.2	8.2	5.1	5.1	3.7	3.8	0.9	1.3	0.6	0.6	0.6	0.6	0.7	0.7	0.8	0.8	0.6	1.7	0.8	2.2
34	21	1.1	1.1	2.0	0.7	1.2	1.2	1.4	1.4	1.0	1.0	0.9	0.9	0.9	0.9						
36	1	8.1	8.1	5.2	5.3	3.2	3.2	0.6	0.7	0.5	0.6	0.4	0.7	0.5	0.8	0.6	1.0	0.7	1.4	0.9	1.7
36	21	1.0	1.4	0.9	0.9	1.3	1.3	1.5	1.5	1.0	1.0	0.8	0.8	0.7	0.7	0.7	0.7				

Table 8 – $\zeta = 1.0$

MATRIX ELEMENTS = -DLOG10((EXACT - COMPUTED)/EXACT); IMAGINARY PARTS OF POLYRT COMPUTED ROOTS

		(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)	(11)	(12)	(13)	(14)	(15)	(16)	(17)	(18)	(19)	(20)
4	1	15.7	15.7	7.4	7.4																
6	1	15.7	15.7	15.7	15.7	15.7	15.7														
8	1	15.7	15.7	15.7	15.7	15.7	15.7	3.5	3.5												
10	1	15.7	15.7	15.7	15.7	15.7	15.7	4.9	4.9	4.9	4.9										
12	1	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7								
14	1	15.7	15.7	15.7	15.7	4.2	4.2	3.6	3.6	3.3	3.3	3.4	3.4	4.0	4.0						
16	1	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7				
18	1	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	3.0	3.0	2.2	2.2	2.7	2.7		
20	1	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	1.3	1.3	1.2	1.2	1.4	1.4	2.1	2.1
22	1	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	1.4	1.4	0.7	0.7	0.5	0.5	0.6	0.6	0.9	0.9
22	21	1.6	1.6																		
24	1	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	1.1	1.1	0.2	0.2	0.1	0.1	0.0	0.0	0.0	0.0	0.1
24	21	0.1	0.5	0.5	15.7																
26	1	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	0.8	0.8	0.1	0.1	0.0	0.0	0.0	0.0	0.0	0.0
26	21	0.0	0.0	0.3	0.3	15.7	15.7														
28	1	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	0.2	0.2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
28	21	0.0	0.0	0.0	0.0	0.0	0.3	0.3	15.7												
30	1	15.7	15.7	15.7	15.7	15.7	15.7	15.7	0.6	0.6	0.1	0.1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	15.7	0.0	0.0
30	21	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.3	0.3										
32	1	15.7	15.7	15.7	15.7	15.7	15.7	15.7	0.6	0.6	0.1	0.1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
32	21	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	15.7								
34	1	15.7	15.7	15.7	15.7	15.7	15.7	15.7	15.7	0.0	0.0	0.0	0.0	0.0	0.0	15.7	0.0	0.0	0.0	0.0	0.0
34	21	0.0	0.0	0.0	15.7	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0						
36	1	15.7	15.7	15.7	15.7	1.7	1.7	15.7	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
36	21	0.0	15.7	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0				

Tables 7 and 8 present results associated with equation 6 for $\zeta = 1.0$. These results are intended to provide data at the limit of the breakdown of computation, that is, when roots are exactly coincident in the complex plane, or when damping is critical.

Data for over-damped systems is not presented, since all roots would then become real, and data presented in Tables 1 and 2 would then become relevant to this situation.

In summary, the data presented shows that polynomials in power series form are adequate for low- to moderate-order oscillatory systems which do not have any near-critically damped system frequencies. For oscillatory systems with nearly critically damped system frequencies and for non-oscillatory systems, a yellow caution flag is raised. The disturbing aspect of the data derived is that numerical accuracy degrades gradually. Recognizing that users always attempt to push methods a little too far, it is inevitable that if a change in approach is not made, false conclusions will inevitably result.

SYSTEM EQUATIONS, PRIMARY DATA

Specification of a mathematical simulation model begins at the point at which all input parameters are physically measurable items of data (e.g., length, mass, feedback gain, position, orientation). The computing problems which lie along the path from this starting point to the need to evaluate transfer functions in the complex frequency domain must now be examined.

The most extensively used representation for a feedback control system is the block diagram. The parameters defined in each block of the block diagram, before reduction to canonical form, can normally be considered as "primary data," or physically measurable items. From the block diagram it is possible to set up both time and frequency domain equations which characterize system response.

If the system is continuous, a system of first order ordinary differential equations of the form:

$$\dot{X} = AX + BU \tag{8}$$

$$Y = CX \tag{9}$$

can be directly set up, where X are state variables, Y and U are output and input variables, and the coefficient matrices A, B, and C have all elements defined in terms of primary data. If the system is discrete, a system of first order finite difference equations of the form:

$$X(k+1) = AX(k) + BU(k) \tag{10}$$

$$Y(k) = CX(k) \tag{11}$$

can be directly set up with all matrix elements defined in terms of primary data.

Sampled data systems do not lead as readily to continuous or discrete time equations. They can, however, be defined in terms of linear mixed operation equations (see Reference 10) with all matrix elements defined in terms of primary data. From this set of mixed operation equations it is possible to set up, via a straightforward procedure, a system of first order finite difference equations of the form given by equations 10 and 11. This set-up procedure, however, requires the computation of both a matrix exponential and an integral involving the matrix exponential. A detailed study of the condition and of the relevant associated computational operations has been done by van Loan, (see References 22, 30, 31, and 32). With respect to computational error, if subroutine PADE (Reference 32) is used, then the user can define on input an acceptable bound for the perturbation matrix E. The algorithm uses this data to set control parameters which guarantee that results are exact for some matrix A + E. With respect to condition, van Loan (see Reference 30) concludes that "the matrix exponential problem is relatively well-conditioned when A [in e^{At}] is a normal matrix and more poorly-conditioned where A has a defective eigensystem." These results do not answer the important question of whether poorly-conditioned eigensystems imply sensitivity of the exponential. This is an area for future research.

One way to get a "feel" for whether or not A is poorly conditioned with respect to exponentiation is to attempt to find the nonsingular transformation matrix ϕ which will reduce it to block diagonal form (see method 18 in Reference 22). This can be done via subroutine BDIAG, developed and given in Reference 1, or by its extended version, namely BLOCK IT which is discussed in Reference 9. If the resultant blocking is poor, that is if the algorithm yields blocks much greater than one or two and ϕ has a large condition number, with respect to matrix inversion, "beware."

Once the feedback control equations are defined in the form given either by equations 8 and 9 or equations 10 and 11, numerous methods exist for obtaining transfer functions as polynomial ratios in factored polynomial form. In a relative sense, poles are easy to obtain. They are simply the eigenvalues of A which are obtainable via application of the EISPACK algorithms (Reference 27). Computation of zeros is considerably more difficult, as proven by the profusion of "better" methods in the literature, (see References 3, 4, 5, 6, 17, 18, 20, 23 and 26).

If the objective is the computation of transmission zeros; i.e., in a rough sense those complex frequencies at which transmission through the system is blocked, then application of the QZ-method is currently the best available (see Reference 18). For a definition of the QZ-method, see References 21 and 29, and for its subroutine code, see Reference 11.

If the objective is the computation of the zeros for particular transfer functions in the transfer function matrix, any of the referenced methods can be used if the system is of low order. Caution must be exercised if the system is of moderate to large order. Davison, in Reference 5, page 481,

cautions against the use of Brockett's method (Reference 4). Davison's method (Reference 5) is unsatisfactory because it is based upon introducing an arbitrarily large number into a matrix, the eigenvalues of which are the sought-after zeros. This approach has the net effect of increasing the norm of the matrix and hence the error associated with follow-on eigenvalue computation. Eigenvalues, computed via the QR method are exact for some matrix A + E where the norm of the perturbation matrix E is proportional to the norm of A (see Reference 36, pages 353 and 367).

Sandberg and So's method (Reference 26) relies on a correct determination of numerator degree. This is a difficult numerical task for moderate- to large-order problems; an incorrect determination leads to incorrect results. Bollinger's method (Reference 3), Marshall's method (Reference 20), and Patel's method (Reference 23) each rely upon the computation of the roots of a polynomial in power series form. This, for moderate- to large-order systems, is an ill-conditioned computing problem. Kaufman's method (Reference 17) is based upon solving the generalized eigenvalue problem and can be readily implemented by use of the QZ-algorithm rather than Kaufman's algorithm. If the QZ-algorithm is used results are exact for the problem $(A + E) X = \lambda (B + F) X$ where bounds in both E and F can be defined. In application, the only problem the author has encountered with this method is that eigenvalues known a priori to be identically equal to zero are computed as "small non-zero numbers." For the problem of simulating moderate- to large-order low frequency systems these small numbers tend to get mixed up with true small magnitude eigenvalues. This is a problem with the QZ-algorithm, however, and not with Kaufman's method. Perhaps this problem can be corrected by prefacing the entrance into the QZ-method with a prebalancing step analogous to the prebalancing step associated with the QR-method and implemented via subroutine BALANC (see Reference 27). The goal would be to isolate eigenvalues which can be determined via zero error row-column permutations. In application, this is extremely useful since the eigenvalues which are isolated have zero computational error. Frequently these are the zero frequencies which, if computed, become troublesome small magnitude complex numbers. This is an area of future study.

It is the author's opinion that for moderate to large order systems the zero-finding algorithm NUMS contained in the DISCOS program (Reference 2, page 65, Vol. I and page B-141, Vol. II) is the best available. It does not have any of the disadvantages cited above and has been successfully applied on some very difficult poorly-conditioned problems for which the other methods above have been less than satisfactory. An upgraded version of NUMS is currently in the DISCOS program being distributed by COSMIC. An equivalent version can be easily created by simply using EISPACK (Reference 27) and LINPACK (Reference 8) based subroutines where required. These were not available when NUMS was originally coded.

The preceding remarks pertain when the so-called "state variable" approach is utilized. In general application it is often desirable to obtain feedback control transfer functions via block diagram algebra (see Reference 7). If this approach is applied, the net effect is to arrive at transfer functions defined as ratios of polynomials defined in power series form. These are normally obtained via a sequence of algebraic operations with functions of polynomials. As previously stated, finding the zeros of a moderate- to large-degree polynomial in power series form is frequently an ill-conditioned computing problem and should be avoided.

If all blocks of the block diagram can be defined as low-order polynomial ratios in either the Laplace variable s or the Z-transform variable z then it is possible to write directly from the block diagram a higher order state variable equation of the form:

$$P(s)\,Y(s) = BU(s) \tag{12}$$

where the elements of the square matrix P (s) are ratios of low-degree polynomials in power series form and U (s) and Y (s) are input and output variables.

If the feedback system is sampled data, then a mixed operation higher order state variable equation can be written of the form:

$$\begin{bmatrix} P_{11}(s) & P_{12}(z) \\ P_{21}(s) & P_{22}(z) \end{bmatrix} \begin{Bmatrix} Y_1(s) \\ Y_2(z) \end{Bmatrix} = \begin{bmatrix} B_{11} \\ 0 \end{bmatrix} \left\{ U(z) \right\} \tag{13}$$

where again matrix elements are ratios of low-degree polynomials. In this case input must be assumed constant over the sampling interval, hence U (z) and not U (s).

At present, there exists no general-purpose algorithm to go directly from equation 13 to an equation of the form:

$$P(z)\,Y(z) = BU(z) \tag{14}$$

However, it appears that a combination of the ideas presented in References 10 and 38 will lead to the desired transformation. This is an area of future research.

In order to obtain the elements of the transfer function matrix G (z) or G (s) an inverse of a matrix of polynomials in s or z must be computed. That is:

$$G(z) = P(z)^{-1} \tag{15}$$

On the bottom line this calls for the need to compute the zeros of a series of determinantal equations, in which matrix elements are low-order polynomials in power series form. It should be noted that, at this point, polynomial coefficients can still be considered as "primary data."

Let the matrix of rational polynomials, for which the determinant must be computed, be given by:

$$\frac{H(z)}{d(z)} \tag{16}$$

where d (z) is the polynomial which is the least common denominator and H (z) is a square matrix of low-degree polynomials in power series form. If r is the degree of the highest degree polynomial in H (z), then it is rather straightforward to write:

$$\det [H(z)] = \det [z^r C_r + z^{r-1} C_{r-1} + \cdots + C_0] = 0 \tag{17}$$

where the order of each of the matrices $C_0, C_1, \cdots, C_r$ equals N, the order of H (z). Moler and Stewart (Reference 21) point out that this is the "λ-matrix" problem. On page 242 they show that it is solvable via the QZ-algorithm. Their approach, however, demands that a generalized eigenvalue problem of order N(r + 1) be solved. In application, it is frequently possible to set up a generalized eigenvalue problem of considerably lower order, as follows.

Step 1. Scan each column of the matrix H (z) and record the degree of the highest degree polynomial. Let:

r_i = degree of highest degree polynomial in column i;

if $r_i = 0$, reset it equal to 1.

Step 2. View z as a differential operator and consider the set of transform equations:

$$\begin{bmatrix} H_{11}(z) & \cdots & H_{1N}(z) \\ \cdot & & \cdot \\ \cdot & & \cdot \\ \cdot & & \cdot \\ H_{N1}(z) & \cdots & H_{NN}(z) \end{bmatrix} \begin{Bmatrix} X_1(z) \\ \cdot \\ \cdot \\ \cdot \\ X_N(z) \end{Bmatrix} = \begin{Bmatrix} 0 \\ \cdot \\ \cdot \\ \cdot \\ 0 \end{Bmatrix} \tag{18}$$

where each of the state variables in X (z) are higher order state variables. This equation can be rewritten as a set of first order state variable equation of the form:

$$A\,X(z) + zBX(z) = 0 \tag{19}$$

where the order of system of equations is $R = r_1 + r_2 + \cdots + r_N$. This algorithm is best illustrated by a simple example. Consider the system:

$$\begin{bmatrix} a_3 z^3 + a_2 z^2 + a_0 & 0 & b_2 z^2 + b_0 \\ c_4 z^4 + c_3 z^3 & d_0 & z \\ e_2 z^2 + e_1 z + e_0 & f_0 & g_5 z^5 + g_4 z^4 + g_0 \end{bmatrix} \begin{Bmatrix} X_1(z) \\ X_2(z) \\ X_3(z) \end{Bmatrix} = \begin{Bmatrix} 0 \\ 0 \\ 0 \end{Bmatrix} \tag{20}$$

Let:

$$X_{11}(z) = X_1(z) \qquad X_{31}(z) = X_3(z)$$

$$X_{12}(z) = zX_{11}(z) \qquad X_{32}(z) = zX_{31}(z)$$

$$X_{13}(z) = zX_{12}(z) \qquad X_{33}(z) = zX_{32}(z)$$

$$X_{14}(z) = zX_{13}(z) \qquad X_{34}(z) = zX_{33}(z)$$

$$X_{35}(z) = zX_{34}(z)$$

$$X_{21}(z) = X_2(z) \tag{21}$$

Directly substitute into equation 32 to obtain:

$$\left[\begin{array}{cccc|c|cccccc}
-1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & a_3 & 0 & 0 & 0 & b_2 & 0 & 0 & 0 \\
\hline
0 & 0 & 0 & c_4 & 0 & 1 & 0 & 0 & 0 & 0 \\
\hline
0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 \\
0 & e_2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & g_5
\end{array}\right]
\left\{\begin{array}{c}
zX_{11}(z) \\ zX_{12}(z) \\ zX_{13}(z) \\ zX_{14}(z) \\ \hline zX_{21}(z) \\ \hline zX_{31}(z) \\ zX_{32}(z) \\ zX_{33}(z) \\ zX_{34}(z) \\ zX_{35}(z)
\end{array}\right\}$$

$$+ \left[\begin{array}{cccc|c|cccccc}
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
a_0 & 0 & a_2 & 0 & 0 & b_0 & 0 & 0 & 0 & 0 \\
\hline
0 & 0 & 0 & c_3 & d_0 & 0 & 0 & 0 & 0 & 0 \\
\hline
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
e_0 & e_1 & 0 & 0 & f_0 & g_0 & 0 & 0 & g_4 & 0
\end{array}\right]
\left\{\begin{array}{c}
X_{11}(z) \\ X_{12}(z) \\ X_{13}(z) \\ X_{14}(z) \\ \hline X_{21}(z) \\ \hline X_{31}(z) \\ X_{32}(z) \\ X_{33}(z) \\ X_{34}(z) \\ X_{35}(z)
\end{array}\right\}
= \left\{\begin{array}{c}
0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0
\end{array}\right\} \qquad (22)$$

The important point to note here is that we have arrived at the need for solving a generalized eigenvalue problem via a trivial zero-computational-error operation.

The eigenvalues of equation 22, which is of the form of equation 19, are the values of z which satisfy the determinantal equation 17:

$$\det [H(z)] = 0 \qquad (17)$$

These can then be used to rewrite this equation in the desired factor polynomial form:

$$\det [H(z)] = G^*(z - z_1)(z - z_2) \cdots (z - z_M) = 0 \qquad (23)$$

Where $M \leq R$ and where G^* is obtained by evaluating both sides of equation 23 for any value of z not equal to an eigenvalue and solving directly for G^*.

M is determined by examining the computed values of z_j; j = 1, 2, · · ·, R. M is equal to the number of computed eigenvalues "not" at numerical infinity in the complex plane (see Reference 21, page 241).

REDUCED ORDER SYSTEM EQUATIONS

In References 1 and 9, it is shown that a computational algorithm has been developed which can be used to compute the transformation matrix ϕ which will reduce a real nonsymmetric matrix A to block diagonal form. This can be expressed as:

$$\phi^{-1} A\phi = \text{diag}\,(B_1, B_2, \cdots, B_k) \tag{24}$$

where k is the number of blocks. The transformation matrix is guaranteed to be nonsingular, and all elements of it are real. Since each block is quasi-upper triangular, usually 1 × 1 or 2 × 2, each block and their associated columns of ϕ can be interpreted as system frequencies and quasi-system modes. In brief consider the problem:

$$\dot{X} = AX + BU \tag{25}$$

Find ϕ and define the coordinate transformation:

$$X = \phi q \tag{26}$$

Direct substitution leads to:

$$\dot{q} = \phi^{-1} A\phi q + \phi^{-1} BU \tag{27}$$

or by equation 24.

$$\dot{q} = \text{diag}\,(B_1, B_2, \cdots, B_K)\, q + \phi^{-1}\, BU \tag{28}$$

Transform the equations and obtain:

$$q(s) = [s\,\square - \text{diag}\,(B_1, \cdots, B_K)]^{-1}\, \phi^{-1}\, BU\,(s) \tag{29}$$

where $\square$ is the unit diagonal of appropriate dimension.

Then back transform and obtain

$$X(s) = \phi\, \text{diag}\,[(s\,\square - B_1)^{-1}, (s\,\square - B_2)^{-1}, \cdots, (s\,\square - B_K)^{-1}]\, \phi^{-1}\, BU\,(s) \tag{30}$$

where partial inversion of the block diagonal matrix has been carried out.

Making use of the fact that the elements of ϕ define system mode observability and that the elements of ϕ^{-1} define system mode controllability, it is possible to utilize engineering judgement to write reduced order transfer functions as partial fraction expansions of the form

$$G(s) = \sum_{j=1}^{L} \frac{N_j(s)}{D_j(s)} \tag{31}$$

where $L \leqslant K$.

Normally, the degree of all $D_j(s)$ is one or two with the degree of $N_j(s)$ being less than or equal to the degree of $D_j(s)$. Larger order blocks lead to larger degree polynomials, which can be determined via the "method of Leverrier" presented first in 1840 (Reference 14, page 166). The method is outlined in Appendix A. When using the method of Leverrier it is essential to bear in mind that this method is not recommended for "large matrices," since it requires too many operations and tends to be

numerically unstable (Reference 14, page 149). The method of Leverrier is applicable, however, for the determination of the matrix of polynomials defined by:

$$(s\,\square - B_j)^{-1}$$

due to the fact that each matrix B_j is of low order. If the matrix B_j is moderate to large order, blocking is by definition "poor." This normally implies that the basic computing problem is ill-conditioned. This does not imply that the physical system is ill-conditioned. It does, however, imply that the analyst, through poor engineering judgment, has set up a set of system dynamics equations which defy solution. A few common examples of poor engineering judgment are:

- An attempt to model system response frequencies which differ by many orders of magnitude;
- A poor choice of physical units. Ideally, all numbers of significance in the computing problem should be of comparable order; and
- A poor choice of state variables such as, for example, using a Euler angle sequence which is subject to "gimbal lock," using relative rather than absolute coordinates for rate and/or displacement measures or vice versa.

When blocking is poor, "stop." The analyst is advised to step back and to reexamine the basic formulation which led to the troublesome computing problem. It is usually a much simpler problem to reduce frequency spectrum, change units, or use a different set of state variables than to fight an ill-conditioned computing problem.

Again, the objective is to write the transfer function G (s) given in equation 31 as a truncated partial fraction expansion as a ratio of polynomials both given in factored polynomial form. The denominator is the least common denominator; since all B_i in equation 29 are quasi-upper triangular it is simply the root product of all system frequencies not truncated out. The computation of the numerator polynomial, in factored polynomial form, can be computed via the QZ-algorithm by recognition of the fact that:

$$\det \begin{bmatrix} 0 & D_2(s) & D_3(s) & D_L(s) & 1 \\ D_1(s) & 0 & D_3(s) & D_L(s) & 1 \\ D_1(s) & D_2(s) & 0 & D_L(s) & 1 \\ . & . & . & . & . \\ . & . & . & . & . \\ . & . & . & . & . \\ D_1(s) & D_2(s) & D_3(s) & 0 & 1 \\ N_1(s) & N_2(s) & N_3(s) & N_L(s) & 0 \end{bmatrix} \qquad (32)$$

equals the numerator polynomial exactly. But, this is simply a determinantal equation involving a matrix of polynomials and hence is solvable via the QZ-method and algorithm presented in the previous section. The validity of the fact that the determinant defined by equation 32 equals the numerator polynomial for the partial fraction expansion exactly defined by equation 31 can be easily verified by expanding the determinant in terms of the cofactors of $N_1(s), N_2(s), \cdots, N_L(s)$.

CONCLUSION

An attempt has been made to summarize a large body of numerical error analysis literature in a language familiar to controls engineers. If your computing problems are of moderate-to large-order, beware. The algorithms that you now use with confidence for low-order systems may not be applicable.

REFERENCES

1) Bavely, C. A., and Stewart, G. W., "An Algorithm for Computing Reducing Subspaces by Block Diagonalization," *SIAM Journal of Numerical Analysis*, Vol. 16, April 1979, pp. 359-367.

2) Bodly, C. S., Devers, A. D., Park, A. C., and Frisch, H. P., "A Digital Computer Program for the Dynamic Interaction Simulation of Controls and Structure (DISCOS)," NASA Technical Paper 1219, Vols. 1 and 2, May 1978 (program code available COSMIC, University of Georgia, Athens, Georgia, Program GSC-12422).

3) Bollinger, K. E. and Mathur, J. C., "To Compute the Zeros of Large Systems," *IEEE Trans. Automatic Control*, February 1971, pp. 95-96.

4) Brockett, R. W., "Poles, Zeros and Feedback: State Space Interpretation," *IEEE Trans. Automatic Control*, AC-10, April 1965, pp. 129-135.

5) Davison, E. J., "A Computational Method for Finding the Zeros of a Multivariable Linear Time Invariant System," *Automatica*, Vol. 6, 1970, pp. 481-484.

6) Davison, F. J., and Wang, S. H., "Properties and Calculation of Transmission Zeros of Linear Multivariable Systems," *Automatica*, Vol. 10, 1974, pp. 643-658.

7) DiStefana, J. J., Stubberud, A. R., and Williams, I. J., "Feedback and Control Systems," *Schaum's Outline Series*, McGraw-Hill Book Co., 1967.

8) Dongarra, J. J., Bunch, J. R., Moler, C. B., and Stewart, G. W., "LINPACK User's Guide," *SIAM Review*, 1979.

9) Frisch, H. P., "Reduced Order Feedback Control Equations for Linear Time and Frequency Domain Analysis," NASA Technical Paper 1818, January 1981.

10) Frisch, H. P., "Time and Frequency Domain Analysis of Sampled Data Controllers via Mixed Operation Equations," NASA Technical Paper 1817, October 1980.

11) Garbow, B. S., Boyle, J. M., Dongarra, J. J., and Moler, C. B., *Matrix Eigensystem Routine -- EISPACK Guide Extension*, Vol. 51, Lecture Notes in Computer Science, Springer-Verlag, 1977.

12) Gautschi, W., "On the Condition of Algebraic Equations," *Numerische Mathematik*, Vol. 21, 1973, pp. 405-424.

13) Givens, J. W., "Numerical Computation of the Characteristic Values of a Real Symmetric Matrix," Report ORNL-1574, Oak Ridge National Laboratory, Oak Ridge, Tennessee, 1954.

14) Householder, A. S., *The Theory of Matrices in Numerical Analysis*, Blaisdell Publishing Company, 1964.

15) Jenkins, M. A., "Algorithm 493: Zeros of a Real Polynomial," *ACM Trans. Mathematical Software*, Vol. 1, June 1975, pp. 178-189.

16) Jenkins, M. A., and Traub, J. F., "A Three-Stage Algorithm for Real Polynomials Using Quadratic Iteration," *SIAM Journal of Numerical Analysis*, Vol. 7, 1970, pp. 545-566.

17) Kaufman, I., "On Poles and Zeros of Linear Systems," *IEEE Trans. on Circuit Theory*, Vol. CT-20, No. 2, March 1973, pp. 93-101.

18) Laub, A. J., and Moore, B. C., "Calculation of Transmission Zeros Using QZ-Techniques," *Automatica*, Vol. 14, 1978, pp. 557-566.

19) MacFarlane, A. G. J., *Frequency-Response Methods in Control Systems*, IEEE Press, 1979.

20) Marshall, S. A., "Remarks on Computing the Zeros of Large Systems," *IEEE Trans. Automatic Control*, April 1972, p. 261.

21) Moler, C. B., and Stewart, G. W., "An Algorithm for Generalized Matrix Eigenvalue Problems," *SIAM Journal of Numerical Analysis*, Vol. 10, No. 2, April 1973, pp. 241-256.

22) Moler, C., and van Loan, C., "Nineteen Dubious Ways to Compute the Exponential of a Matrix," *SIAM Review*, Vol. 20, No. 4, October 1978, pp. 801-836.

23) Patel, R. V., "On the Computation of Numerators of Transfer Functions of Linear Systems," *IEEE Trans. Automatic Control*, August 1973, pp. 400-401.

24) Rice, J. R., "On the Conditioning of Polynomial and Rational Forms," *Numerische Mathematik*, Vol. 7, 1965, pp. 426-435.

25) Rice, J. R., "A Theory of Condition," *SIAM Journal of Numerical Analysis*, Vol. 3, No. 2, June 1966, pp. 287-310.

26) Sandberg, I. W., and So, H. C., "A Two-Sets-of-Eigenvalues Approach to the Computer Analysis of Linear Systems," *IEEE Trans. Circuit Theory*, CT-16, November 1969, pp. 509-517.

27) Smith, B. T., Boyle, J. M., Dongarra, J. J., Garbow, B. S., Ikebe, Y., Klema, V. C., and Moler, C. B., *Matrix Eigensystem Routines – EISPACK Guide*, Vol. 6, Springer-Verlag, 1976.

28) Stewart, G. W., *Introduction to Matrix Computation*, Academic Press, 1973.

29) Stewart, G. W., "On the Sensitivity of the Eigenvalue Problem Ax = λBx," *SIAM Journal of Numerical Analysis*, Vol. 9, No. 4, December 1972, pp. 669-686.

30) van Loan, C., "The Sensitivity of the Matrix Exponential," *SIAM Journal of Numerical Analysis*, Vol. 14, No. 6, December 1977, pp. 971-981.

31) van Loan, C., "Computing Integrals Involving the Matrix Exponential," *IEEE Trans. Automatic Control*, Vol. AC-23, No. 3, 1978 (no subroutine source code).

32) van Loan, C., "Computing Integrals Involving the Matrix Exponential," Department of Computer Science, Cornell University, Ithaca, N.Y., TR 76-298 (with subroutine source code).

33) von Neumann, J., and Goldstine, H. H., "Numerical Inverting of Matrices of High Order," *Bulletin of the American Mathematical Society*, Vol. 53, 1947, pp. 1021-1099.

34) Wilkinson, J. H. "Modern Error Analysis," *SIAM Review*, Vol. 13, No. 4, October 1971, pp. 548-568.

35) Wilkinson, J. H., *Rounding Errors in Algebraic Processes*, Prentice Hall Inc., 1963.

36) Wilkinson, J. H., and Reinsch, C., "Handbook for Automatic Computation," Vol. II, *Linear Algebra*, Springer-Verlag, 1971.

37) Wilkinson, J. H., *The Algebraic Eigenvalue Problem*, Oxford, 1965.

38) Wilkinson, J. H., "The Differential System $B\dot{x} = Ax$ and the Generalized Eigenvalue Problem $Au = \lambda Bu$," National Physical Laboratory, England, NPL Report NAC 73, January 1977.

Blank Page

APPENDIX A

THE METHOD OF LEVERRIER

Blank Page

THE METHOD OF LEVERRIER

Use of good engineering judgment normally leads to a well-conditioned computing problem with good blocking characteristics. Consider the need to compute the matrix $(s\square - B)^{-1}$, where B is assumed low order, equal to n. The method of Leverrier is given as follows (see Reference 14, page 166):

Step 1. Write the adjoint of the matrix $(S\square - B)$ as:

$$(s\square - B)^{A} = C_0 s^{n-1} - C_1 s^{n-2} + \cdots \pm C_{n-1}$$

Step 2. Make use of the equations

$$C_0 = \square$$

$$C_1 = \gamma_1\square - C_0 B$$

$$C_2 = \gamma_2\square - C_1 B$$

$$C_{n-1} = \gamma_{n-1}\square - C_{n-2} B$$

$$C_{n-1} B = \gamma_n \square$$

and

$$\text{trace } [C_j B] = (j+1)\gamma_{j+1} \qquad j = 0, 1, \ldots, n-1$$

to compute all coefficient matrices $C_0, C_1, \ldots, C_{n-1}$. This can be done iteratively by first using C_0 to compute γ_1, then C_1, γ_2, C_2, et cetera.

Step 3. Make use of the fact that all B are quasi-upper triangular, and let:

$$s_1, s_2, \cdots, s_n$$

be the n eigenvalues of B. The desired matrix inverse is thus given by:

$$(s\square - B)^{-1} = \frac{C_0 s^{n-1} - C_1 s^{n-2} + \cdots \pm C_{n-1}}{(s - s_1)(s - s_2)\cdots(s - s_n)}$$

Again, all B are of low order and hence the power series form of polynomials is acceptable.

1. Report No. NASA TP-1814	2. Government Accession No.	3. Recipient's Catalog No.
4. Title and Subtitle Pitfalls and Guidelines for the Numerical Evaluation of Moderate-Order System Frequency Response		5. Report Date June 1981 6. Performing Organization Code 712
7. Author(s) Harold P. Frisch		8. Performing Organization Report No. 81F0042
9. Performing Organization Name and Address Goddard Space Flight Center Greenbelt, Maryland 20771		10. Work Unit No. 11. Contract or Grant No. 13. Type of Report and Period Covered Technical Paper
12. Sponsoring Agency Name and Address National Aeronautics and Space Administration Washington, D.C. 20546		14. Sponsoring Agency Code
15. Supplementary Notes		

16. Abstract

The design and evaluation of a feedback control system via frequency response methods relies heavily upon numerical methods. In application, one can usually develop low-order simulation models which for the most part are devoid of numerical problems. However, when complex feedback interactions, for example, between instrument control systems and their flexible mounting structure, must be evaluated, simulation models become moderate to large order and numerical problems become common. The fundamental reason for this is that as system order enters the range of about 20 to 200, many popular tried and true methods are subject to severe numerical problems. An attempt is made in this paper to summarize a large body of relevant numerical error analysis literature in a language understandable to nonspecialists. The intent is to provide engineers using simulation models with an "engineering feel" for potential numerical problems without getting intertwined in the complexities of the associated mathematical theory. Guidelines are also provided by suggesting alternate state-of-the-art methods which have good numerical evaluation characteristics.

17. Key Words (Selected by Author(s))	18. Distribution Statement
Analytical and Numerical Methods, Guidance and Control, Spacecraft Simulation	Unclassified – Unlimited Subject Category 18

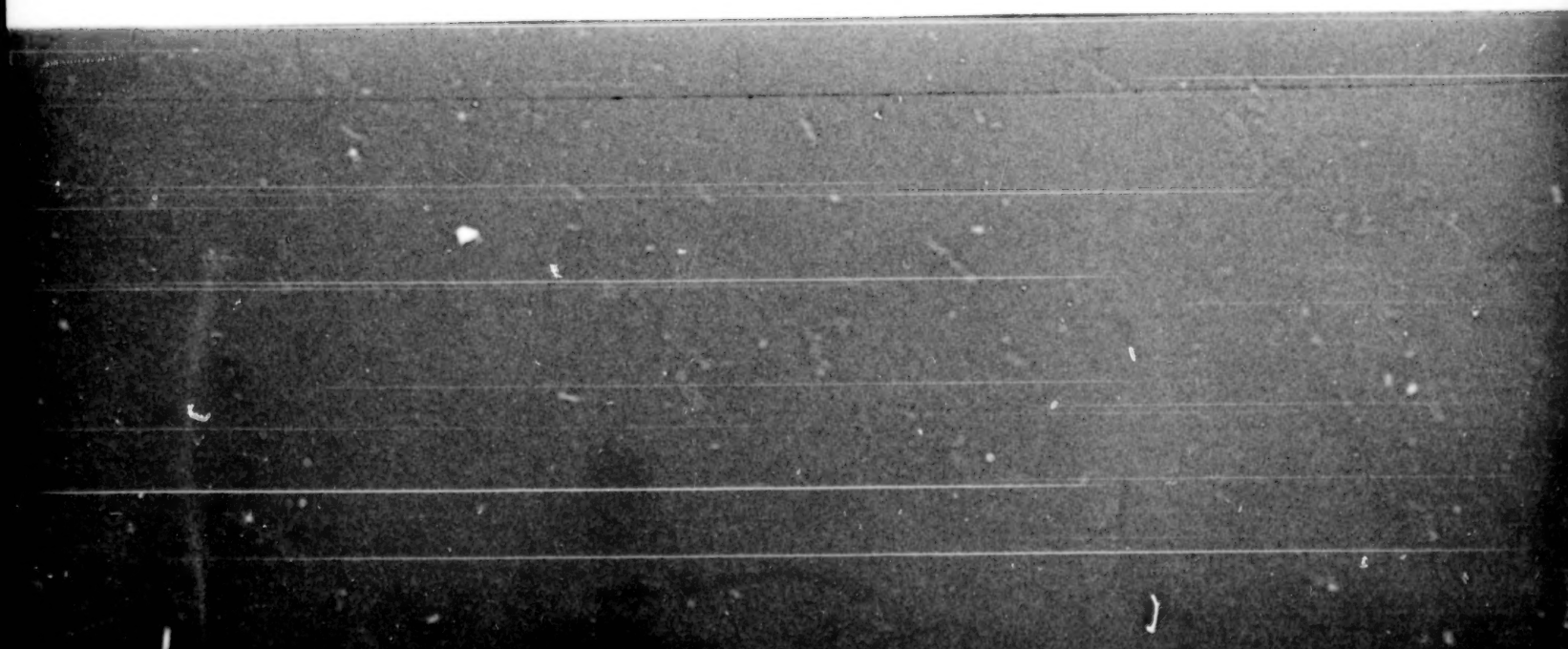
19. Security Classif. (of this report)	20. Security Classif. (of this page)	21. No. of Pages	22. Price*
Unclassified	Unclassified	37	A03

*For sale by the National Technical Information Service, Springfield, Virginia 22161

A-4

NASA-Langley, 1981

90%



END

10-16-81